基于故障注入的可靠性评估研究平台综述

胡瑜 李晓维

摘要：可靠性评估研究平台的主要目的是快速准确地评估各类故障对计算系统可靠性的影响，以便为系统设计人员提供定量分析数据。本文概述了计算系统的可靠性评估方法，着重介绍了国内外在基于故障注入的可靠性评估平台方面开展的代表性工作，同时介绍了本课题组在相关研究工作上的进展。

关键词：高端计算系统、可靠性评估、故障注入

1　引言

随着高端计算系统（High-End Computing System）采用数以万计的高性能处理器，利用海量并行获得每秒百万亿次甚至千万亿次浮点运算能力（PetaFlops/s）的峰值性能，系统平均无故障时间（Mean Time Between Failure，简称 MTBT）也随着系统硬件规模的日益扩大而不断下降。图 1 显示了单个部件每小时失效概率为 0.0001, 0.00001 和 0.000001（即 MTBF 为 10^4, 10^5 和 10^6 小时）的情况下，系统的 MTBF 值随节点数量增加而不断下降[1]。从图中可见，对于有 20 万个节点的 P 级计算系统，其 MTBF 仅有几个小时。另一方面，随着集成电路工艺特征尺寸的缩小、电源电压的降低和频率的升高，集成电路芯片对于电压扰动、电磁干扰以及辐射等各种噪声干扰变得更加敏感。上述两方面因素综合作用，导致可靠性问题成为高端计算系统面临的六大严峻挑战之一[2]。搭建可靠性评估研究平台的主要目的就是快速准确地评估各类故障对高端计算系统可靠性的影响，以便为系统设计人员提供定量分析数据。

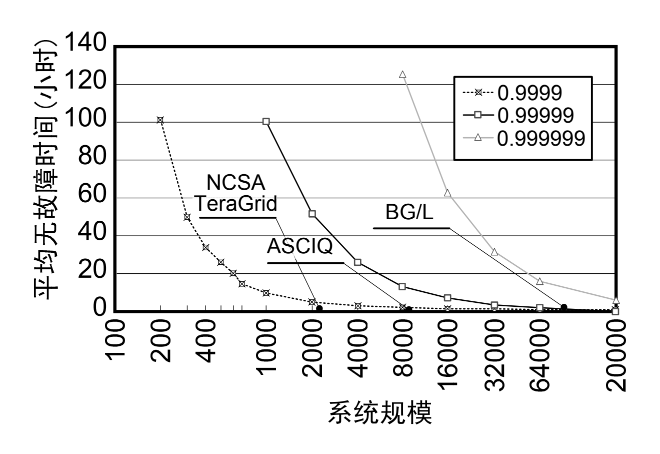


图1. 高端计算系统的平均无故障时间[1]

计算系统的可靠性评估方法包括基于测量的方法、基于解析模型的方法和基于故障注入的方法。基于测量的方法是指在运行实际工作负载的过程中，测量实际系统自然出现差错时的行为，因而可以获得真实的数据。但是由于实际运行过程中发生差错的频率很低，因此基于测量的方法往往需要很长的时间才能获得足够的数据进行统计分析。基于解析模型的方法是指建立计算系统的数学模型，例如马尔科夫链模型和 Petri 网模型，通过数学模型计算出相关指标。然而模型自身和输入参数的不准确有可能导致分析结果出现严重偏差。故障注入是指按照预先选定的故障模型，采用某种策略将故障人为地引入到运行特定工作负载的目标系统中，并且观察和分析引入故障后系统的行为，从而获得定性或者定量结果的实验过程。作为评估计算系统可靠性的一种主要方法，故障注入技术的提出始见于二十世纪七十年代初期 IBM 公司的内部技术报告[3]，之后被工业界用于容错计算系统的可靠性评估，在八十年代中期受到高校和研究部门的关注，目前已在各种计算系统的可靠性评估中得到广泛应用。相比于测量和解析模型方法，故障注入方法更为经济灵活，在可靠性评估中占有越来越重要的地位，因此本文将着重介绍国内外在基于故障注入的可靠性评估平台方面开展的工作。

2 基于故障注入的可靠性评估平台概述

基于故障注入的可靠性评估平台主要包括目标系统以及故障注入、故障库、负载生成器、负载库、控制器、监测器、数据采集器与数据分析器等模块[4]，如图 2 所示：

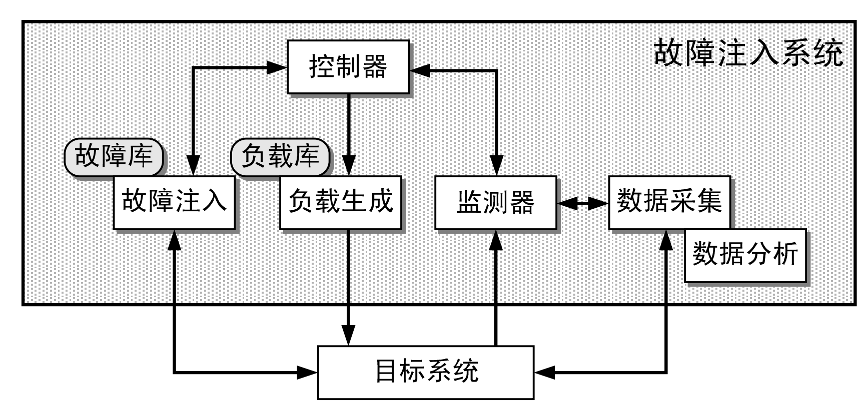


图2. 基于故障注入的可靠性评估平台的框架

故障注入模块负责将故障注入到目标系统中；监测器负责跟踪故障注入模块执行命令的情况并在必要的时候启动数据采集器。数据采集器在线地收集数据，而数据分析器离线地处理并分析数据。由控制器负责控制整个实验过程。故障类型可以是固定型故障（Stuck-At Fault）、位翻转故障（Bit-flip Fault）、桥接故障（Bridging Fault）、杂散电流（Spurious Current）和电压浪涌（Power Surge）等。

根据注入故障的实现方式，故障注入方法主要分为通过硬件实现的故障注入（Hardware-Implemented Fault Injection，简称 HWIFI）、通过软件实现的故障注入（Software-Implemented Fault Injection，简称 SWIFI）和基于模拟的故障注入（Simulation-Based Fault Injection）。

通过硬件实现的故障注入是指利用额外的硬件将故障引入目标系统的硬件中。按照故障注入器是否直接接触目标系统，硬件实现的故障注入可进一步分为接触式和非接触式两类。顾名思义，接触式硬件实现的故障注入是利用管脚级探针[5][6]、夹具[29]或者插座[7]，直接在目标芯片上产生电压或者电流变化。非接触式硬件实现的故障注入是指利用重离子辐射[8][9][10]、电磁效应[11]、激光[12]或者扫描链[13]，在目标芯片上产生杂散电流。硬件实现故障注入方法的优点是可以在目标系统的任意位置产生故障，缺点是难以精确控制故障注入的时间和位置，并且有可能损坏目标系统。

通过软件实现的故障注入方法是指在编译时或者运行时修改程序[14][15] [16][17][18][19]，使目标系统的正常状态在程序执行过程中发生改变。软件实现的故障注入不仅能够对应用程序注入故障，还能对操作系统注入故障。其优点是易于实现且成本低，缺点是由于不能够将故障注入到软件不可访问的位置，软件实现的故障注入只能够部分模拟实际的故障情况；时间精度比较低，不适合模拟诸如总线和处理器故障等潜伏期较长的故障；需要修改应用程序，因此有可能使工作负载发生变化。

基于模拟的故障注入是指在系统设计阶段，向用 VHDL[1]或者 Verilog 硬件描述语言设计的系统注入故障，可以在开关级[20]、门级和寄存器传输级[21][22][30]、行为级[23][24] [25]等不同抽象层次上进行。基于模拟的故障注入方法具有良好的可控性和可观性，能够在系统设计的早

[1] VHSIC Hardware Description Language，甚高速集成电路硬件描述语言

期阶段进行可靠性评估，有助于设计人员尽早采取适当的容错措施提升系统的可靠性。缺点是抽象层次越低的故障注入模拟实验，虽然模拟精度越高，但速度越慢，获得有意义的统计数据所需要的时间越长。近年来有研究人员利用故障精简来加速模拟[26]，以及采用 FPGA[2]仿真技术来进行可靠性评估 [27][28]。

在下一节中，我们将选取一些具有代表性的研究工作，详细介绍基于故障注入的可靠性评估平台。

3　代表性研究工作

3.1 AFIT (Advanced Fault Injection Tool) [6]

AFIT 是由西班牙瓦伦西亚大学（Universitat de València）开发的一个硬件实现的故障注入平台，采用夹具对目标芯片进行管脚级故障注入。图 3 显示了 AFIT 的总体框架，包括：用作控制器的个人计算机、同步与触发模块、定时模块、目标系统激活模块、事件读取模块、高速故障注入模块和目标系统原型。

同步与触发模块的作用是控制故障注入实验的起始时间；定时模块的作用是为高速故障注入模块提供 40MHz 的时钟和故障注入使能信号 AI，而 AI 信号的波形随注入故障的数量和类型的变化而变化；目标系统激活模块的作用是初始化目标系统原型；事件读取模块利用计数器和踪迹存储器决定什么时候读取系统的响应。高速故障注入模块的结构如图 4 所示，包括注入激活逻辑和有效差错检测器。在接收到来自定时模块输出的 AI 信号后，注入激活逻辑就连通晶体管。当连接供电电源的晶体管被连通时，通过 I_{OUT} 输出信号向原型系统注入逻辑值为 1 的故障；当接地的晶体管被连通时，通过 I_{OUT} 向原型系统注入逻辑值为 0 的故障。有效差错检测器的作用是设置连接到个人计算机的有效差错存储器 MEE（memory of effective error）信号，当 I_{OUT} 确实改变了原型系统管脚的逻辑值时，MEE 为 1，表明故障注入成功，否则为 0 表明故障注入不成功。AFIT 能够以 40MHz 的频率向目标系统注入故障，并且能够选择是注入瞬态故障、间歇故障还是永久故障，故障发生的位置、持续的时间与频率均可控。其中，瞬态故障持续时间控制范围在 100ns 到 4μs 之间，间歇故障持续时间控制范围为 100ns 至 2μs，间隔控制范围 1 至 65ms，永久故障持续时间为 1.2s。

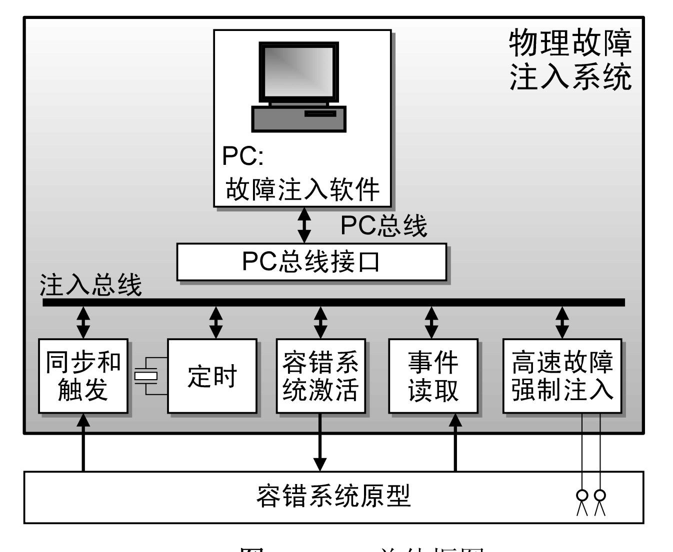



图3. AFIT 总体框图

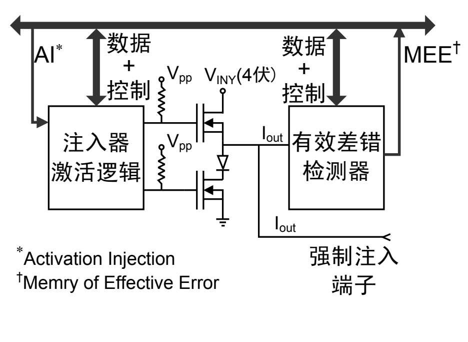


图4. 高速注入模块结构

[2] Field Programmable Gate Array 现场可编程门阵列

3.2 Ftape (Fault Tolerance and Performance Evaluator) [17]

Ftape 是由美国伊利诺伊大学香槟分校（University of Illinois at Urbana-Champaign）开发的一个软件实现的故障注入平台。图 5 显示了 Ftape 的总体框架。注入故障的位置包括软件可访问的 CPU 寄存器、内存和磁盘子系统，故障类型为单个或者多个位翻转、置位或者复位。注入故障的时刻和位置既可以按照某种概率分布函数（例如指数分布或者正态分布）来随机生成，也可以根据负载的特点来生成（例如将故障注入到某些使用率比较高的部件，以加速故障效应）。其中磁盘系统的故障注入是通过运行驱动程序中的一段代码来实现的，例如总线故障或者定时器故障，因此没有额外的硬件开销。Ftape 已用于测量两个 Tandem 容错计算机原型的可靠性以及发生故障时系统的性能降级情况。

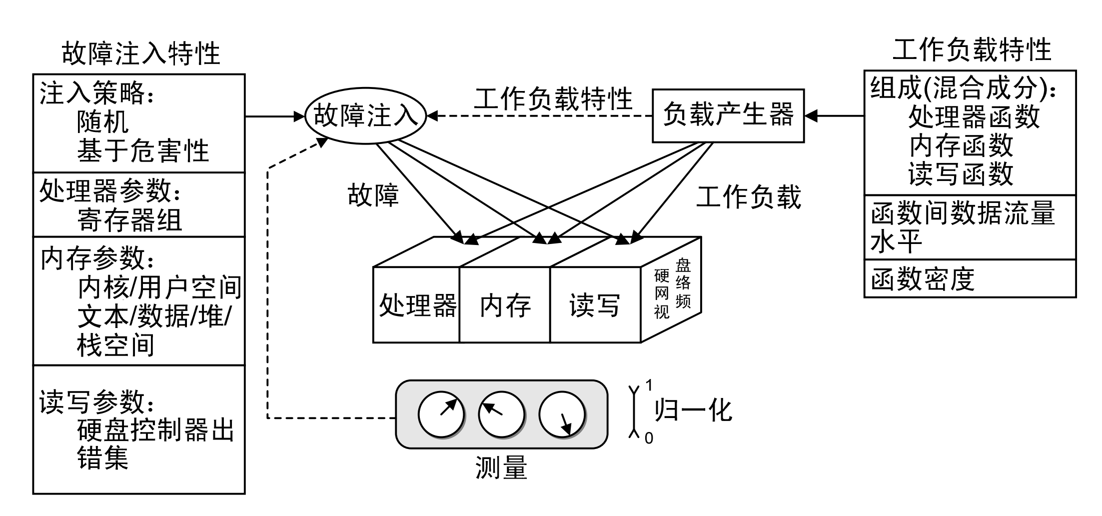


图5. Ftape 总体框架

3.3 DEPEND [24]

DEPEND 是由美国伊利诺伊大学香槟分校开发的另一个可靠性评估平台，采用了基于模拟的故障注入方法，由相互通信的进程集合描述计算系统的行为，故障模型为功能故障。DEPEND 的总体框图如图 6 所示。使用时首先用 DEPEND 库里的对象编写 C++控制程序，然后编译链接为运行时环境，并且进行故障注入。此后启动修复，并生成统计数据的报告。DEPEND 库里的对象包括活动部件（模拟基本的服务器，提供先到先服务、轮叫（Roundrobin）服务策略，提供手动故障注入与修复），故障注入器（按照设定的概率分布或负载特点注入故障），校验和（计算校验和），故障报告器（收集故障统计数据、显示 MTBF、MTBR[3]、可用性与覆盖率，提供各个故障的具体信

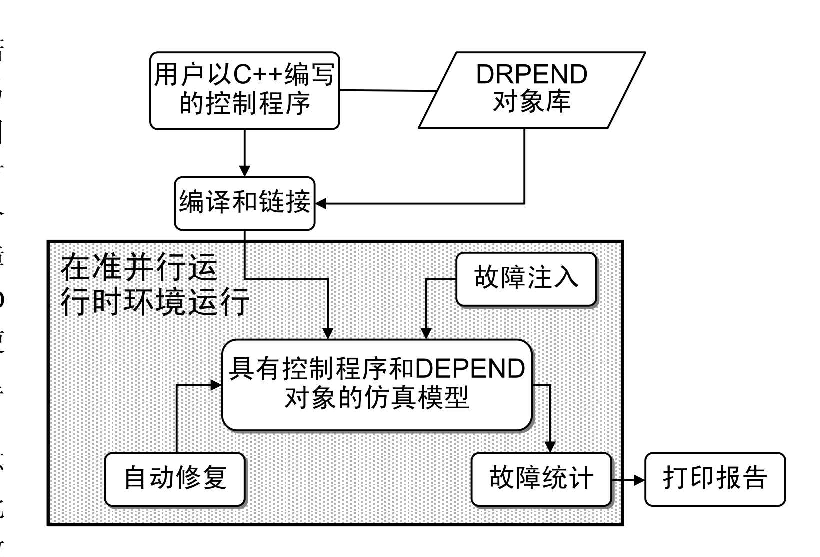


图6. DEPEND 总体框架

[3] Mean Time between Repairs，平均修复间隔时间

息），表决器（模拟一个带有超时功能的基本表决器，允许用户自定义表决策略），服务器（具有活动部件的属性，但模拟带有冗余部件的服务器。提供自动注入故障，自动修复与重构功能），链接（模拟通信通道，支持链接故障、包出错、丢包以及用户定义的故障，支持自动重试），多模冗余器（模拟双机互测、三模冗余和 N 模冗余）和故障管理器（记录故障并关闭超出故障阈值的部件）。

3.4 FuSE (Fault injection using SEmulation) [28]

FuSE 是由奥地利维也纳技术大学（Vienna University of Technology）开发的一个基于现场可编程门阵列（FPGA）仿真的可靠性评估平台，用于提高故障注入模拟实验的速度，其总体框图如图 7 所示。其中 SEmulator 引擎支持三种模式：模拟模式——与传统的硬件描述语言模拟一样，测试用例（Testbench）和被测设计（DUT）都运行在主机上，模拟速度大约为每秒数千时钟节拍；协同模拟模式——被测的全部设计或者部分设计被下载到 HMX2-AS2 FPGA 开发板中，因此将在读写管理（IO Manager）的控制下，通过现场可编程门阵列的 PCIexpress 接口把测试用例输出的信号输入到被仿真的电路中，同时把被观测的信号送回主机，仿真速度最高可达到每秒 20 万时钟节拍；时钟加速模式——以最高速度在现场可编程门阵列中进行仿真，由于在测试用例和被测设计之间没有交互，因此被测设计的内部信号状态不可观测，仿真速度最高可达 100MHz。

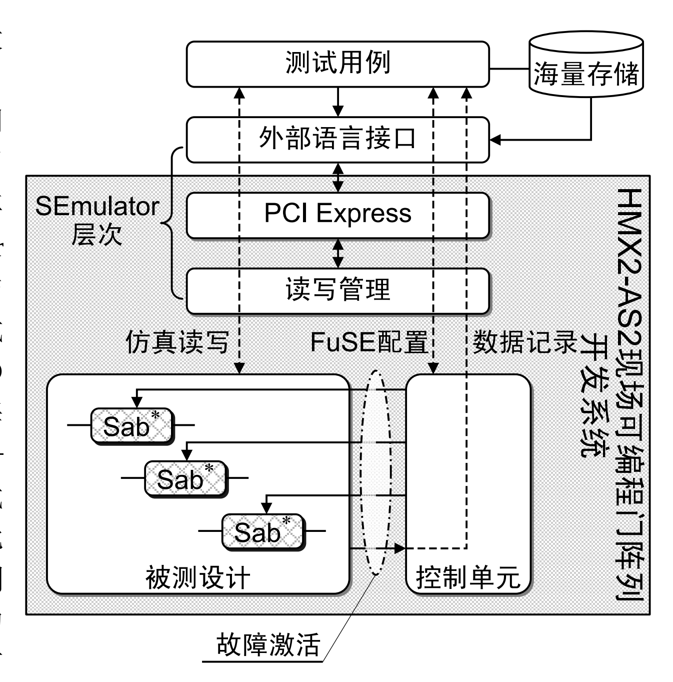


图7. FuSE 总体框架

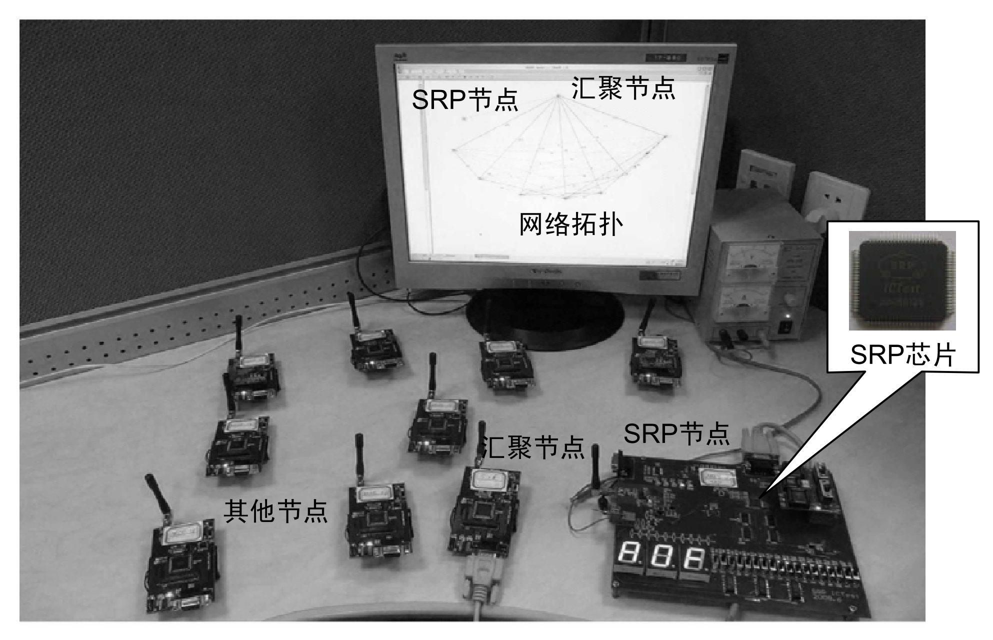
4 我们的工作

为提高微处理器的容错能力，我们在一款低功耗微处理器中实现了基于自测试（Built-In Self-Test, BIST）、自诊断（Built-In Self-Diagnosis, BISD）和自修复（Built-In Self-Repair, BISR）（简称 3S）的可靠性设计，该芯片采用 0.18μm 的中芯国际（SMIC）标准工艺库投片生产。我们采用了基于模拟的故障注入方法来检验该处理器的容错能力，特色是将故障注入逻辑设计在处理器中，因此无论是在 RTL[4]级、门级还是封装后的芯片，都可以由用户自定义注入故障的数量、位置和时刻，并且最多可向处理器的 SRAM[5]中注入 20 个故障字，故障类型为固定型故障。为实现计算机与处理器芯片中故障注入模块、内建自测试和内建自修复模块的通信和控制，我们设计了一个控制电路来实现在测试与修复模式或故障注入模式下内部电路与外部计算机的通信，并制定了相应的通信协议。

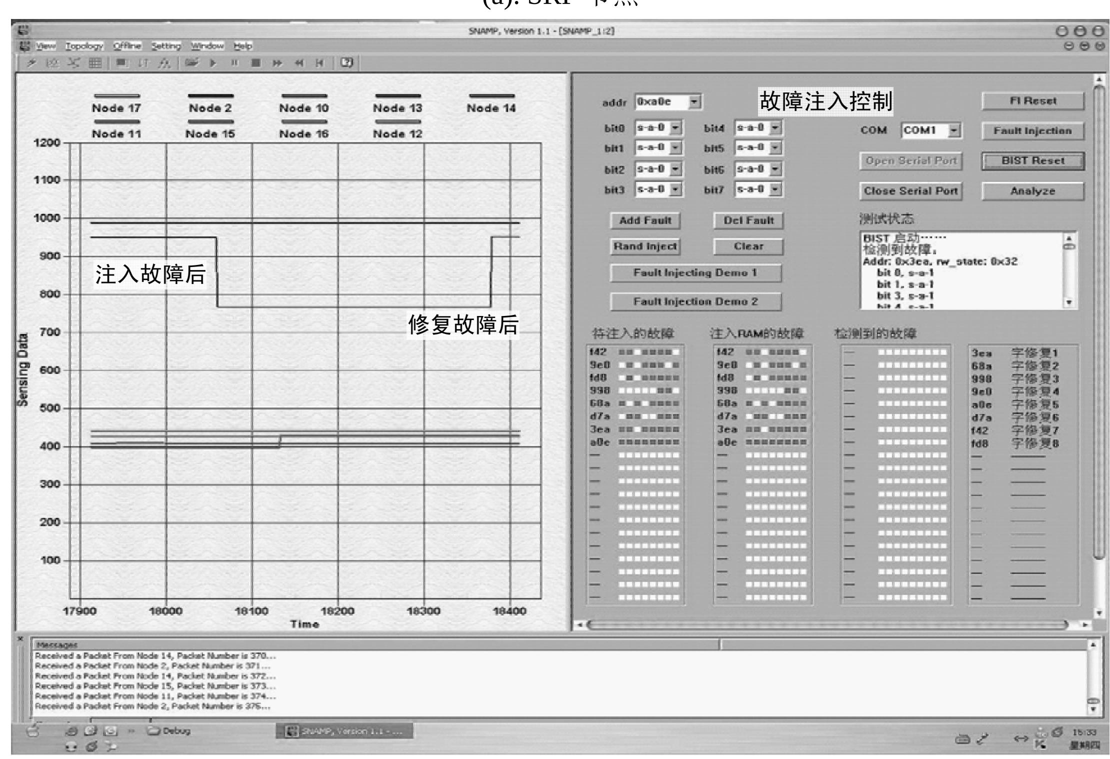
如图 8(a)所示的可靠性评估平台包括一个 SRP[6]传感节点、四个普通的传感节点和一个

[4] Register Transfer Level，寄存器传输级
[5] Static Random Access Memory，静态随机存储器
[6] Self-Repairable Microprocessor，自修复微处理器

数据采集节点。工作负载为无线传感网络应用。由 SRP 传感节点和普通的传感节点收集的光强数据通过数据采集节点传输到主机的控制软件进行分析和显示。图 8(b)显示了在 SRP 中注入 8 个故障后，SRP 通过自测试自诊断自修复，最终又恢复正常工作的过程。



(a). SRP 节点



(b) SRP 节点的故障注入、自测试、自诊断与自修复过程

图8. SRP 处理器在无线传感网络节点中的应用

5 结束语

本文综述了国内外在基于故障注入的可靠性评估平台方面开展的研究，分别介绍了基于硬件实现的故障注入、软件实现的故障注入、基于模拟和基于 FPGA 仿真的四项有代表性的

工作，并介绍了我们在一款具有 3S 能力的处理器设计中，利用基于模拟的故障注入进行的可靠性评估工作。如何针对多核处理器以及高端计算系统开发相应的可靠性评估研究平台，这是我们未来将要开展的工作。

参考文献

[1] Daniel A. Reed, Charng-da Lu, Celso L. Mendes, Big Systems and Big Reliability Challenges, *Proceedings of Parallel Computing: Software Technology, Algorithms, Architectures and Applications (PARCO)*, 2003, pp. 729-736.

[2] Carl G. Tengwall, IBM®Blue Gene®/P - An Overview of a Petaflop Capable System, *http://www.nsc.liu.se/lcsc2007/presentations/LCSC_2007-tengwall.pdf*.

[3] Harlan D. Mills, On the Statistical Validation of Computer Programs, *Technical Report, FSC-72-6015*, IBM Federal Systems Division, 1972.

[4] Mei-Chen Hsueh, Timothy K. Tsai, Ravishankar K. Iyer, Fault Injection Techniques and Tools, *IEEE Computer*, 1997, 30(4): 75-82.

[5] Henrique Madeira, Mário Zenha Rela, Francisco Moreira, João Gabriel Silva, RIFLE: A General Purpose Pin-level Fault Injector, *Proceedings of European Dependable Computing Conference on Dependable Computing (EDCC)*, 1994, pp. 199-216.

[6] J. R. Martínez, P.J. Gil, G. Martín, C. Pérez, J. J. Serrano, Experimental Validation of High-Speed Fault-Tolerant Systems Using Physical Fault Injection”, *Proceedings of International Working Conference on Dependable Computing for Critical Applications (DCCA)*, 1999, pp. 233-249.

[7] J. Arlat, Y. Crouzet, and J.C. Laprie, Fault Injection for Dependability Validation of Fault-Tolerant Computer Systems, *Proceedings of International Symposium on Fault-Tolerant Computing (FTCS)*, 1989, pp. 348-355.

[8] U. Gunneflo, J. Karlsson, J. Torin, Evaluation of error detection schemes using fault injection by heavy-ion radiation, *Proceedings of International Symposium on Fault-Tolerant Computing (FTCS)*, 1989, pp. 340-347.

[9] Cristian Constantinescu, Neutron SER Characterization of Microprocessors, *Proceedings of International Conference on Dependable Systems and Networks (DSN)*, 2005, pp. 754-759.

[10] Jeffrey Kellington, Ryan McBeth, Pia Sanda and Ronald Kalla, IBM® POWER6™ Processor Soft Error Tolerance Analysis Using Proton Irradiation, *Proceedings of Workshop on Silicon Errors in Logic-Systems Effects (SELSE)*, 2007.

[11] J. Karlsson, J. Arlat, and G. Leber, Application of Three Physical Fault Injection Techniques to the Experimental Assessment of the MARS Architecture, *Proceedings of International Working Conference on Dependable Computing for Critical Applications (DCCA)*, 1995, pp. 150-161.

[12] [Sampson1998] J. R. Sampson, W. Moreno, F. Falquez, A technique for automatic validation of fault tolerant designs using laser fault injection, *Proceedings of International Symposium on Fault-Tolerant Computing (FTCS)*, 1998, pp. 162-167.

[13] P. Folkesson, S. Svensson, J. Karlsson, A Comparison of Simulation Based and Scan Chain Implemented Fault Injection, *Proceedings of International Symposium on Fault-Tolerant Computing (FTCS)*, 1998, pp. 284-293.

[14] J.H. Barton, E.W. Czeck, Z.Z. Segall, and D.P. Siewiorek, Fault Injection Experiments Using FIAT, *IEEE Transactions on Computers*, 1990, 39(4): 575-582.

[15] [Kanawati1995] G.A. Kanawati, N.A. Kanawati, and J.A. Abraham, FERRARI: A Flexible Software-Based Fault and Error Injection System, *IEEE Transactions on Computers*, 1995, 44(2): 248-260.

[16] S. Han, K.G. Shin, and H.A. Rosenberg, Doctor: An Integrated Software Fault-Injection Environment

for Distributed Real-Time Systems, *Proceedings of International Computer Performance and Dependability Symposium (IPDS)*, 1995, pp. 204-213.

[17] T.K. Tsai and R.K. Iyer, An Approach to Benchmarking of Fault-Tolerant Commercial Systems, *Proceedings of International Symposium on Fault-Tolerant Computing (FTCS)*, 1996, pp. 314-323.

[18] [Carreira1998] J. Carreira, H. Madeira, J.G. Silva, Xception: A technique for the experimental evaluation of dependability in modern computers, *IEEE Trans. on Software Engineering*, 1998, 24(2): 125-136.

[19] A. Baldini, A. Benso, S. Chiusano, P. Prinetto, “BOND”: An interposition agents based fault injector for Windows NT, *Proceedings of International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT)*, 2000, pp. 387-395.

[20] H.R. Zarandi, G. Miremadi, and A. Ejlali, Fault Injection into Verilog Models for Dependability Evaluation of Digital Systems, *Proceedings of International Symposium on Parallel and Distributed Computing (ISPDC)*, 2003, pp. 281-287.

[21] V. Sieh, O. Tschäche, and F. Balbach, VERIFY: Evaluation of reliability using VHDL-models with embedded fault descriptions, *Proceedings of International Symposium on Fault-Tolerant Computing (FTCS)*, 1997, pp. 32-36.

[22] D. Gil, J. Gracia, J. C. Baraza, and P. J. Gil, A study of the effects of transient fault injection into the VHDL model of a fault-tolerant microcomputer system, *Proceedings of International On-Line Testing Workshop (IOLTW)*, 2000, pp. 73-79.

[23] E. Jenn, J. Arlat, M. Rimén, J. Ohlsson, and J. Karlsson, “Fault injection into VHDL models: The MEFISTO tool, *Proceedings of International Symposium on Fault-Tolerant Computing (FTCS)*, 1994, pp. 356-363.

[24] [Goswami1997] Kumar K. Goswami, Ravishankar K. Iyer, and Luke Young, DEPEND: A Simulation-Based Environment for System Level Dependability Analysis, *IEEE Transactions on Computers*, 1997, 46(1): 60-74.

[25] Juan-Carlos Baraza, Joaquín Gracia, Sara Blanc, Daniel Gil, and Pedro-J. Gil, Enhancement of Fault Injection Techniques Based on the Modification of VHDL Code, *IEEE Transactions. on VLSI Systems*, 2008, 16(6): 693-706.

[26] L. Berrojo, F. Corno, L. Entrena, I. González, C. López, M. Sonza, G. Squillero, An Industrial Environment for High-Level Fault-Tolerant Structures Insertion and Validation, *Proceedings of VLSI Test Symposium (VTS)*, 2002, pp. 229-236.

[27] P. L. Civera, L. Macchiarulo, M. Rebaudengo, M. Sonza Reorda, M. Violante, Exploiting FPGA for accelerating Fault Injection Experiments, *Proceedings of International On-Line Testing Workshop (IOLTW)*, 2001. pp. 9-13.

[28] Marcus Jeitler, Martin Delvai, Stefan Reichör, FuSE - A Hardware Accelerated HDL Fault Injection Tool, *Proceedings of Southern Conference on Programmable Logic (SPL)*, 2009, pp. 89-94.

[29] 王建莹，孙峻朝，容错计算机系统可靠性评估工具：HFI-2 故障注入器，电子学报， 1999，27(11): 24-26.

[30] 黄海林，唐志敏，许彤，龙芯 1 号处理器的故障注入方法与软错误敏感性分析，计算机研究与发展，2006，43(10): 1820-1827.

作者简介：

胡　瑜　中国科学院计算技术研究所副研究员，硕士生导师，huyu@ict.ac.cn

李晓维　中国科学院计算技术研究所研究员、博士生导师、中国科学院计算机系统结构重点实验室副主任，中国计算机学会理事、容错计算专业委员会主任、JCST 副主编